FINNEGAN, HENDERSON, FARABOW, GARRETT & DUNNER, L.L.P.
1300 I STREET, N. W.
WASHINGTON, DC 20005-3315

202 • 408 • 4000
FACSIMILE 202 • 408 • 4400

WRITER'S DIRECT DIAL NUMBER:

(202) 408-4376

ATLANTA
404 • 653 • 6400
PALO ALTO
650 • 849 • 6600

TOKYO
011 • 813 • 3431 • 6943
BRUSSELS
011 • 322 • 646 • 0353

December 10, 1999

ATTORNEY DOCKET NO. **06502.0289**

Box PATENT APPLICATION
Assistant Commissioner for Patents
Washington, D.C. 20231

Re: New U.S. Patent Application
Title: SYSTEM AND METHOD FOR ENABLING SCALABLE SECURITY IN A VIRTUAL PRIVATE NETWORK
Inventors: Germano CARONNI et al.

Sir:

We enclose the following papers for filing in the United States Patent and Trademark Office in connection with the above patent application.

1. Application - 41 pages, including 7 independent claims and 45 claims total.

2. Drawings - 11 sheets of formal drawings (Figures 1-10).

This application is being filed under the provisions of 37 C.F.R. § §1.53(b) and 1.53(f). Applicants await notification from the Patent and Trademark Office of the time set for filing the Declaration.

Please accord this application a serial number and filing date.

No filing fee is being paid at this time. With the exception of the filing fee, the Commissioner is hereby authorized to charge any additional filing fees due and any other fees due under 37 C.F.R. § 1.16 or § 1.17 during the pendency of this application to our Deposit Account No. 06-0916.

Respectfully submitted,

FINNEGAN, HENDERSON, FARABOW,
GARRETT & DUNNER, L.L.P.

By: _____
Matthew A. Kaminer
Reg. No. 44,379

MAK:js
Enclosures

Attorney Docket No.: 06502.0289

UNITED STATES PATENT APPLICATION

FOR

SYSTEM AND METHOD FOR ENABLING SCALABLE SECURITY IN A VIRTUAL PRIVATE NETWORK

BY

GERMANO CARONNI

AMIT GUPTA

SANDEEP KUMAR

TOM R. MARKSON

CHRISTOPH L. SCHUBA

GLENN C. SCOTT

RELATED APPLICATIONS

The following identified U.S. patent applications are relied upon and are incorporated by reference in this application.

U.S. Patent Application No. _____, entitled "SYSTEM AND METHOD FOR SEPARATING ADDRESSES FROM THE DELIVERY SCHEME IN A VIRTUAL PRIVATE NETWORK," bearing attorney docket no. 06502.0280, and filed on the same date herewith.

U.S. Patent Application No. _____, entitled "TRULY ANONYMOUS COMMUNICATIONS USING SUPERNETS WITH THE PROVISION OF TOPOLOGY HIDING," bearing attorney docket no. 06502.0281, and filed on the same date herewith.

U.S. Patent Application No. _____, entitled "METHOD AND SYSTEM FOR FACILITATING RELOCATION OF DEVICES ON A NETWORK," bearing attorney docket no. 06502.0283, and filed on the same date herewith.

U.S. Patent Application No. _____, entitled "SANDBOXING APPLICATIONS IN A PRIVATE NETWORK USING A PUBLIC-NETWORK INFRASTRUCTURE," bearing attorney docket no. 06502.0284, and filed on the same date herewith.

U.S. Patent Application No. _____, entitled "SECURE ADDRESS RESOLUTION FOR A PRIVATE NETWORK USING A PUBLIC NETWORK INFRASTRUCTURE," bearing attorney docket no. 06502.0285, and filed on the same date herewith.

LAW OFFICES
FINNEGAN, HENDERSON, FARABOW, GARRETT, & DUNNER, L.L.P.
1300 I STREET, N.W.
WASHINGTON, D.C. 20005
202-408-4000

U.S. Patent Application No. _____, entitled "DECOUPLING ACCESS CONTROL FROM KEY MANAGEMENT IN A NETWORK," bearing attorney docket no. 06502.0286, and filed on the same date herewith.

U.S. Patent Application No. _____, entitled "CHANNEL-SPECIFIC FILE SYSTEM VIEWS IN A PRIVATE NETWORK USING A PUBLIC NETWORK INFRASTRUCTURE," bearing attorney docket no. 06502.0287, and filed on the same date herewith.

U.S. Patent Application No. _____, entitled "PRIVATE NETWORK USING A PUBLIC-NETWORK INFRASTRUCTURE," bearing attorney docket no. 06502.0288, and filed on the same date herewith.

U.S. Patent Application No. _____, entitled "USING MULTICASTING TO PROVIDE ETHERNET-LIKE COMMUNICATION BEHAVIOR TO SELECTED PEERS ON A NETWORK," bearing attorney docket no. 06502.0290, and filed on the same date herewith.

U.S. Patent Application No. _____, entitled "ANYCASTING IN A PRIVATE NETWORK USING A PUBLIC NETWORK INFRASTRUCTURE," bearing attorney docket no. 06502.0291, and filed on the same date herewith.

U.S. Patent Application No. _____, entitled "SCALABLE SECURITY ASSOCIATIONS FOR GROUPS FOR USE IN A PRIVATE NETWORK USING A PUBLIC-NETWORK INFRASTRUCTURE," bearing attorney docket no. 06502.0292, and filed on the same date herewith.

U.S. Patent Application No. _____, entitled "ENABLING SIMULTANEOUS PROVISION OF INFRASTRUCTURE SERVICES," bearing attorney docket no. 06502.0293, and filed on the same date herewith.

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N. W.
WASHINGTON, D. C. 20005
202-408-4000

FIELD OF THE INVENTION

The present invention relates generally to data processing systems and, more particularly, to a private network using a public-network infrastructure.

BACKGROUND OF THE INVENTION

As part of their day-to-day business, many organizations require an enterprise network, a private network with lease lines, dedicated channels, and network connectivity devices, such as routers, switches, and bridges. These components, collectively known as the network's "infrastructure," are very expensive and require a staff of information technology personnel to maintain them. This maintenance requirement is burdensome on many organizations whose main business is not related to the data processing industry (e.g., a clothing manufacturer) because they are not well suited to handle such data processing needs.

Another drawback to enterprise networks is that they are geographically restrictive. The term "geographically restrictive" refers to the requirement that if a user is not physically located such that they can plug their device directly into the enterprise network, the user cannot typically utilize it. To alleviate the problem of geographic restrictiveness, virtual private networks have been developed.

In a virtual private network (VPN), a remote device or network connected to the Internet may connect to the enterprise network through a firewall. This allows the remote device to access resources on the enterprise network even though it may not be located near any component of the enterprise network. For example, Fig. 1 depicts a VPN 100, where

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N.W.
WASHINGTON, D.C. 20005
202-408-4000

enterprise network 102 is connected to the Internet 104 via firewall 106. By using VPN 100, a remote device D_1 108 may communicate with enterprise network 102 via Internet 104 and firewall 106. Thus, D_1 108 may be plugged into an Internet portal virtually anywhere within the world and make use of the resources on enterprise network 102.

To perform this functionality, D_1 108 utilizes a technique known as tunneling to ensure that the communication between itself and enterprise network 102 is secure in that it cannot be viewed by an interloper. "Tunneling" refers to encapsulating one packet inside another when packets are transferred between two end points (e.g., D_1 108 and VPN software 109 running on firewall 106). The packets may be encrypted at their origin and decrypted at their destination. For example, Fig. 2A depicts a packet 200 with a source Internet protocol (IP) address 202, a destination IP address 204, and data 206. It should be appreciated that packet 200 contains other information not depicted, such as the source and destination port. As shown in Fig. 2B, the tunneling technique forms a new packet 208 out of packet 200 by encrypting it and adding both a new source IP address 210 and a new destination IP address 212. In this manner, the contents of the original packet (i.e., 202, 204, and 206) are not visible to any entity other than the destination. Referring back to Fig. 1, by using tunneling, remote device D_1 108 may communicate and utilize the resources of the enterprise network 102 in a secure manner.

Although VPNs alleviate the problem of geographic restrictiveness, they impose significant processing overhead when two remote devices communicate. For example, if remote device D_1 108 wants to communicate with remote device D_2 110, D_1 sends a packet using tunneling to VPN software 109, where the packet is decrypted and then transferred to

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N. W.
WASHINGTON, D.C. 20005
202-408-4000

the enterprise network 102. Then, the enterprise network 102 sends the packet to VPN software 109, where it is encrypted again and transferred to D_2. Given this processing overhead, it is burdensome for two remote devices to communicate in a VPN environment.

VPNs provide security at the network layer of the OSI model and generally cover all applications. The OSI model is a well-known model used to describe the seven protocol layers in a standard TCP/IP protocol stack. The OSI model contains seven layers that use various forms of control information to communicate with their peer layers in other computer systems. This "blanket security" approach requires the VPN to secure all applications regardless of the individual needs of the application. Because of this drawback VPNs cannot differentiate between security at the application level or security at the node level. Moreover, when communicating between security domains controlled by different VPNs, multiple devices are required to allow the connection, such as firewalls and routers. These devices provide gateway services that enable data to be exchanged between various security domains.

Therefore, it is desirable to provide a dynamic security protocol that easily integrates into existing VPNs.

SUMMARY OF THE INVENTION

Methods and systems consistent with the present invention overcome the shortcomings of existing security protocols by providing dynamic security policies that may change the granularity of the security at the node level, process level, or socket level. Specifically, these shortcomings are met by having a security context that includes a channel number and virtual address associated with each process included in a process table. Since

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N. W.
WASHINGTON, D. C. 20005
202-408-4000

a security policy is required for all processes, secure and insecure processes located on the same channel may communicate with one another. Moreover, processes located on different channels may communicate with one another by a gateway that connects both channels. This scalable blanketing security approach provides an institutionalized method for securing any process, node or socket by providing a unique mechanism for policy enforcement at runtime or by changing the security policies.

In accordance with the purpose of the invention as embodied and broadly described herein, a method provides communication access between a first process and a second process. To provide access, the method appends security context information for the first process in a process table, and opens a socket between the first process and the second process. The method then transmits a packet from the first process to the second process through the open socket. Each packet contains security context information for the first process in the process table.

In another implementation, a method for providing secure communications between a first process and a second process is provided. The method obtains a channel number and a virtual address, and includes the channel number and the virtual address in a field corresponding to the first process in a process table. The method then transmits a datagram that contains the channel number and virtual address from the first process to a socket. The datagram is then received at the second process that contains the channel number and a second virtual address.

In another implementation, a method places processes executed in a node in a security context. The method sends a request from the node to a server to verify a username and a

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N. W.
WASHINGTON, D. C. 20005
202-408-4000

channel identification. In response to the request, the method receives security context information at the node from the server and initiates the process. The security context information includes a virtual address for the node. The method then appends the security context information and the channel identification for the process in a process table that is associated with the process.

BRIEF DESCRIPTION OF THE DRAWINGS

This invention is pointed out with particularity in the appended claims. The above and further advantages of this invention may be better understood by referring to the following description taken in conjunction with the accompanying drawings, in which:

Fig. 1 depicts a conventional virtual private network (VPN) system;

Fig. 2A depicts a conventional network packet;

Fig. 2B depicts the packet of Fig. 2A after it has been encrypted in accordance with a conventional tunneling technique;

Fig. 3 depicts a data processing system suitable for use with methods and systems consistent with the present invention;

Fig. 4 depicts the nodes depicted in Fig. 3 communicating over multiple channels;

Fig. 5 depicts two devices depicted in Fig. 3 in greater detail;

Figs. 6A and 6B depict a flow chart of the steps performed when a node joins a VPN in a manner consistent with the present invention;

Fig. 7 depicts a process table used by the VPN in a manner consistent with the present invention;

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N. W.
WASHINGTON, D.C. 20005
202-408-4000

Fig. 8 depicts a flow chart of the steps performed when sending a packet from a node of the VPN in a manner consistent with the present invention;

Fig. 9 depicts a flow chart of the steps performed when receiving a packet by a node of the VPN in a manner consistent with the present invention; and

Fig. 10 depicts a flow chart of the steps performed when logging out of a VPN in a manner consistent with the present invention.

DETAILED DESCRIPTION

Methods and systems consistent with the present invention provide a "Supernet," which is a private network that uses components from a public-network infrastructure. A Supernet allows an organization to utilize a public-network infrastructure for its enterprise network so that the organization no longer has to maintain a private network infrastructure; instead, the organization may have the infrastructure maintained for them by one or more service providers or other organizations that specialize in such connectivity matters. As such, the burden of maintaining an enterprise network is greatly reduced. Moreover, a Supernet is not geographically restrictive, so a user may plug their device into the Internet from virtually any portal in the world and still be able to use the resources of their private network in a secure and robust manner.

Overview

Fig. 3 depicts a data processing system 300 suitable for use with methods and systems consistent with the present invention. Data processing system 300 comprises a number of

LAW OFFICES
FINNEGAN, HENDERSON, FARABOW, GARRETT, & DUNNER, L.L.P.
1300 I STREET, N.W.
WASHINGTON, D.C. 20005
202-408-4000

devices, such as computers 302-312, connected to a public network, such as the Internet 314. A Supernet's infrastructure uses components from the Internet because devices 302, 304, and 312 contain nodes that together form a Supernet and that communicate by using the infrastructure of the Internet. These nodes 316, 318, 320, and 322 are communicative entities (e.g., processes) running within a particular device and are able to communicate among themselves as well as access the resources of the Supernet in a secure manner. When communicating among themselves, the nodes 316, 318, 320, and 322 serve as end points for the communications, and no other processes or devices that are not part of the Supernet are able to communicate with the Supernet's nodes or utilize the Supernet's resources. The Supernet also includes an administrative node 306 to administer to the needs of the Supernet.

It should be noted that since the nodes of the Supernet rely on the Internet for connectivity, if the device on which a node is running relocates to another geographic location, the device can be plugged into an Internet portal and the node running on that device can quickly resume the use of the resources of the Supernet. It should also be noted that since a Supernet is layered on top of an existing network, it operates independently of the transport layer. Thus, the nodes of a Supernet may communicate over different transports, such as IP, IPX, X.25, or ATM, as well as different physical layers, such as RF communication, cellular communication, satellite links, or land-based links.

As shown in Fig. 4, a Supernet includes a number of channels that its nodes 316-322 can communicate over. A "channel" refers to a collection of virtual links through the public-network infrastructure that connect the nodes on the channel such that only these nodes can communicate over it. A node on a channel may send a message to another node on that

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N. W.
WASHINGTON, D. C. 20005
202-408-4000

channel, known as a unicast message, or it can send a message to all other nodes on that channel, known as a multicast message. For example, channel 1 402 connects node A 316 and node C 320, and channel 2 404 connects node B 318, node C 320, and node D 322. Each Supernet has any number of preconfigured channels over which the nodes on that channel can communicate. In an alternative embodiment, the channels are dynamically defined.

In addition to communication, the channels may be used to share resources. For example, channel 1 402 may be configured to share a file system as part of node C 320 such that node A 316 can utilize the file system of node C in a secure manner. In this case, node C 320 serves as a file system manager by receiving file system requests (e.g., open, close, read, write, etc.) and by satisfying the requests by manipulating a portion of the secondary storage on its local machine. To maintain security, node C 320 stores the data in an encrypted form so that it is unreadable by others. Such security is important because the secondary storage may not be under the control of the owners of the Supernet, but may instead be leased from a service provider. Additionally, channel 2 404 may be configured to share the computing resources of node D 322 such that nodes B 318 and C 320 send code to node D for execution. By using channels in this manner, resources on a public network can be shared in a secure manner.

A Supernet may also contain "linked" channels. These channels are linked by a gateway between the channels. The gateway allows the different channels to communicate with one another. With the gateway, a node on a channel may send a message to another node on a different channel. For example, gateway 410 connects channel 3 406 and channel

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N. W.
WASHINGTON, D. C. 20005
202-408-4000

4 408. Since channels 3 and 4 are linked, node A 316 and node B 318 may communicate with each other using channel 3 406 and channel 4 408.

A Supernet provides a number of features to ensure secure and robust communication among its nodes. First, the system provides authentication and admission control so that nodes become members of the Supernet under strict control to prevent unauthorized access. Second, the Supernet provides communication security services so that the sender of a message is authenticated and communication between end points occurs in a secure manner by using encryption. By providing the security services, the Supernet enables scalable security from the socket level to the node level. Third, the system provides key management to reduce the possibility of an intruder obtaining an encryption key and penetrating a secure communication session. The system does so by providing one key per channel and by changing the key for a channel whenever a node joins or leaves the channel. Alternatively, the system may use a different security policy.

Fourth, the system provides address translation in a transparent manner. Since the Supernet is a private network constructed from the infrastructure of another network, the Supernet has its own internal addressing scheme, separate from the addressing scheme of the underlying public network. Thus, when a packet from a Supernet node is sent to another Supernet node, it travels through the public network. To do so, the Supernet performs address translation from the internal addressing scheme to the public addressing scheme and vice versa. To reduce the complexity of Supernet nodes, system-level components of the Supernet perform this translation on behalf of the individual nodes so that it is transparent to the nodes. Another benefit of the Supernet's addressing is that it uses an IP-based internal

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N.W.
WASHINGTON, D.C. 20005
202-408-4000

addressing scheme so that preexisting programs require little modification to run within a Supernet.

Lastly, the Supernet provides operating system-level enforcement of node compartmentalization in that an operating system-level component treats a Supernet node running on a device differently than it treats other processes on that device. This component (i.e., a security layer in a protocol stack) recognizes that a Supernet node is part of a Supernet, and therefore, it enforces that all communications to and from this node travel through the security infrastructure of the Supernet such that this node can communicate with other members of the Supernet and that non-members of the Supernet cannot access this node. Additionally, this operating system-level enforcement of node compartmentalization allows more than one Supernet node to run on the same machine, regardless of whether the nodes are from the same Supernet, and allows nodes of other networks to run on the same machine as a Supernet node.

Implementation Details

Fig. 5 depicts administrative machine 306 and device 302 in greater detail, although the other devices 304 and 308-312 may contain similar components. Device 302 and administrative machine 306 communicate via Internet 314. Each device contains similar components, including a memory 502, 504; secondary storage 506, 508; a central processing unit (CPU) 510, 512; an input device 514, 516 and a video display 518, 520. One skilled in the art will appreciate that these devices may contain additional or different components.

Memory 504 of administrative machine 306 includes the SASD process 540, VARPD

LAW OFFICES
FINNEGAN, HENDERSON, FARABOW, GARRETT, & DUNNER, L.L.P.
1300 I STREET, N. W.
WASHINGTON, D.C. 20005
202-408-4000

548, and KMS 550 all running in user mode. That is, CPU 512 is capable of running in at least two modes: user mode and kernel mode. When CPU 512 executes programs running in user mode, it prevents them from directly manipulating the hardware components, such as video display 518. On the other hand, when CPU 512 executes programs running in kernel mode, it allows them to manipulate the hardware components. Memory 504 also contains a VARPDB 551 and a TCP/IP protocol stack 552 that are executed by CPU 512 running in kernel mode. TCP/IP protocol stack 552 contains a TCP/UDP layer 554 and an IP layer 556, both of which are standard layers well known to those of ordinary skill in the art. Secondary storage 508 contains a configuration file 558 that stores various configuration-related information (described below) for use by SASD 540.

SASD 540 represents a Supernet: there is one instance of an SASD per Supernet, and it both authenticates nodes and authorizes nodes to join the Supernet. VARPD 548 has an associated component, VARPDB 551, into which it stores mappings of the internal Supernet addresses, known as a node IDs, to the network addresses recognized by the public-network infrastructure, known as the real addresses. The "node ID" may include the following: a Supernet ID (e.g., 0x123), reflecting a unique identifier of the Supernet, and a virtual address, comprising an IP address (e.g., 10.0.0.1). The "real address" is an IP address (e.g., 10.0.0.2) that is globally unique and meaningful to the public-network infrastructure. In a Supernet, one VARPD runs on each machine, and it may play two roles. First, a VARPD may act as a server by storing all address mappings for a particular Supernet into its associated VARPDB. Second, regardless of its role as a server or not, each VARPD assists in address translation for the nodes on its machine. In this role, the VARPD stores into its associated

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N. W.
WASHINGTON, D.C. 20005
202-408-4000

VARPDB the address mappings for its nodes, and if it needs a mapping that it does not have, it will contact the VARPD that acts as the server for the given Supernet to obtain it.

KMS 550 performs key management by generating a new key every time a node joins a channel and by generating a new key every time a node leaves a channel. There is one KMS per channel in a Supernet.

To configure a Supernet, a system administrator creates a configuration file 558 that is used by SASD 540 when starting or reconfiguring a Supernet. This file may specify: (1) the Supernet name, (2) all of the channels in the Supernet, (3) the nodes that communicate over each channel, (4) the address of the KMS for each channel, (5) the address of the VARPD that acts as the server for the Supernet, (6) the user IDs of the users who are authorized to create Supernet nodes, (7) the authentication mechanism to use for each user of each channel, and (8) the encryption algorithm to use for each channel. Although the configuration information is described as being stored in a configuration file, one skilled in the art will appreciate that this information may be retrieved from other sources, such as databases or interactive configurations.

After the configuration file is created, it is used to start a Supernet. For example, when starting a Supernet, the system administrator first starts SASD, which reads the configuration information stored in the configuration file. Then, the administrator starts the VARPD on the administrator's machine, indicating that it will act as the server for the Supernet and also starts the KMS process. After this processing has completed, the Supernet is ready for nodes to join it.

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N. W.
WASHINGTON, D.C. 20005
202-408-4000

Memory 502 of device 302 contains SNlogin script 522, SNlogout script 524, VARPD 526, KMC 528, KMD 530, and node A 522, all running in user mode. Memory 502 also includes TCP/IP protocol stack 534 and VARPDB 536 running in kernel mode.

SNlogin 522 is a script used for logging into a Supernet. Successfully executing this script results in a Unix shell from which programs (e.g., node A 522) can be started to run within the Supernet context, such that address translation and security encapsulation is performed transparently for them and all they can typically access is other nodes on the Supernet. Alternatively, a parameter may be passed into SNlogin 522 that indicates a particular process to be automatically run in a Supernet context. Once a program is running in a Supernet context, all programs spawned by that program also run in the Supernet context, unless explicitly stated otherwise. SNlogout 524 is a script used for logging out of a Supernet. Although both SNlogin 522 and SNlogout 524 are described as being scripts, one skilled in the art will appreciate that their processing may be performed by another form of software. VARPD 526 performs address translation between node IDs and real addresses. KMC 528 is the key management component for each node that receives updates whenever the key for a channel ("the channel key") changes. There is one KMC per node per channel. KMD 530 receives requests from SNSL 542 of the TCP/IP protocol stack 534 when a packet is received and accesses the appropriate KMC for the destination node to retrieve the appropriate key to decrypt the packet. Node A 532 is a Supernet node running in a Supernet context.

TCP/IP protocol stack 534 contains a standard TCP/UDP layer 538, two standard IP layers (an inner IP layer 540 and an outer IP layer 544), and a Supernet security layer (SNSL)

LAW OFFICES
FINNEGAN, HENDERSON, FARABOW, GARRETT, & DUNNER, L.L.P.
1300 I STREET, N. W.
WASHINGTON, D. C. 20005
202-408-4000

542, acting as the conduit for all Supernet communications. To conserve memory, both inner IP layer 540 and outer IP layer 544 may share the same instance of the code of an IP layer. SNSL 542 performs security functionality as well as address translation. It also caches the most recently used channel keys for ten seconds. Thus, when a channel key is needed, SNSL 542 checks its cache first, and if it is not found, it requests KMD 530 to contact the appropriate KMC to retrieve the appropriate channel key. Two IP layers 540, 544 are used in the TCP/IP protocol stack 534 because both the internal addressing scheme and the external addressing scheme are IP-based. Thus, for example, when a packet is sent, inner IP layer 540 receives the packet from TCP/UDP layer 538 and processes the packet with its node ID address before passing it to the SNSL layer 542, which encrypts it, prepends the real source IP address and the real destination IP address, and then passes the encrypted packet to outer IP layer 544 for sending to the destination.

SNSL 542 utilizes VARPDB 536 to perform address translation. VARPDB stores all of the address mappings encountered thus far by SNSL 542. If SNSL 542 requests a mapping that VARPDB 536 does not have, VARPDB communicates with the VARPD 526 on the local machine to obtain the mapping. VARPD 526 will then contact the VARPD that acts as the server for this particular Supernet to obtain it.

Although aspects of the present invention are described as being stored in memory, one skilled in the art will appreciate that these aspects can also be stored on or read from other types of computer-readable media, such as secondary storage devices, like hard disks, floppy disks, or CD-ROM; a carrier wave from a network, such as the Internet; or other forms of RAM or ROM either currently known or later developed. Additionally, although

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N.W.
WASHINGTON, D.C. 20005
202-408-4000

a number of the software components are described as being located on the same machine, one skilled in the art will appreciate that these components may be distributed over a number of machines.

Figs. 6A and 6B depict a flow chart of the steps performed when a node joins a Supernet. The first step performed is that the user invokes the SNlogin script and enters the Supernet name, their user ID, their password, and a requested virtual address (step 602). Of course, this information depends on the particular authentication mechanism used. Upon receiving this information, the SNlogin script performs a handshaking with SASD to authenticate this information. In this step, the user may request a particular virtual address to be used, or alternatively, the SASD may select one for them. Next, if any of the information in step 602 is not validated by SASD (step 604), processing ends. Otherwise, upon successful authentication, SASD creates an address mapping between a node ID and the real address (step 606). In this step, SASD concatenates the Supernet ID with the virtual address to create the node ID, obtains the real address of the SNlogin script by querying network services in a well-known manner, and then registers this information with the VARPD that acts as the server for this Supernet. This VARPD is identified in the configuration file. If the node uses multiple channels to communicate, SASD sends the address mapping to the VARPD that acts as a server for that Supernet.

After creating the address mapping, SASD informs the KMS that there is a new Supernet member that has been authenticated and admitted (step 608). In this step, SASD sends the node ID and the real address to KMS who then generates a key ID, a key for use in communicating between the node's KMC and the KMS ("a node key"), and updates the

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N. W.
WASHINGTON, D.C. 20005
202-408-4000

channel key for use in encrypting traffic on this particular channel (step 610). Additionally, KMS sends the key ID and the node key to SASD and distributes the channel key to all KMCs on the channel as a new key because a node has just been added to the channel. SASD receives the key ID and the node key from KMS and returns it to SNlogin (step 612). After receiving the key ID and the node key from SASD, SNlogin starts a KMC for this node and transmits to the KMC the node ID, the key ID, the node key, the address of the VARPD that acts as the server for this Supernet, and the address of KMS (step 614). The KMC then registers with the KMD indicating the node it is associated with, and KMC registers with KMS for key updates (step 616). When registering with KMS, KMC provides its address so that it can receive updates to the channel key via the Versakey protocol. The Versakey protocol is described in greater detail in IEEE Journal on Selected Areas in Communication, Vol. 17, No. 9, 1999, pp. 1614-1631. After registration, the KMC will receive key updates whenever a channel key changes on one of the channels that the node communicates over.

Next, SNlogin configures SNSL (step 618 in Fig. 6B). In this step, SNlogin indicates which encryption algorithm to use for this channel and which authentication algorithm to use, both of which are received from the configuration file via SASD. SNSL stores this information in an access control list. In accordance with methods and systems consistent with present invention, any of a number of well-known encryption algorithms may be used, including the Data Encryption Standard (DES), Triple-DES, the International Data Encryption Algorithm (IDEA), and the Advanced Encryption Standard (AES). Also, RC2, RC4, and RC5 from RSA Incorporated may be used as well as Blowfish from Counterpane.com. Additionally, in accordance with methods and systems consistent with

LAW OFFICES
FINNEGAN, HENDERSON, FARABOW, GARRETT, & DUNNER, L.L.P.
1300 I STREET, N. W.
WASHINGTON, D.C. 20005
202-408-4000

the present invention, any of a number of well-known authentication algorithms may be used, including Digital Signatures, Kerberos, Secure Socket Layer (SSL), and MD5, which is described in RFC1321 of the Internet Engineering Task Force, April, 1992.

After configuring SNSL, SNlogin invokes an operating system call, SETVIN, to cause the SNlogin script to run in a Supernet context (step 620). In Unix, each process has a data structure known as the "proc structure" that contains the process ID as well as a pointer to a virtual memory description of this process. Fig. 7 depicts a process table 700 that lists all of the proc structures currently executing in memory. The columns 710, 720, 730, and 740 show data regarding the attributes of each process. A record 750 includes for each process: a process ID 710; a process name 720; a Supernet ID 730 indicating the channel the process belongs; and vaddr 740 indicating the virtual address for the node. One skilled in the art will appreciate that process table 700 may contain additional information to maintain the process. To join multiple Supernets, the user repeats the steps of Figs. 6A and 6B for each Supernet.

In accordance with methods and systems consistent with the present invention, the IDs indicating the channels over which the process communicates as well as its virtual address for this process are added to this structure. By associating this information with the process in process table 700, the SNSL layer can enforce that this process runs in a Supernet context. Also during step 620, a gateway may be initiated to communicate across multiple channels. To do so, SNlogin executes a gateway process that spawns two child processes, both of which are connected by a shared-memory region in memory. Each child process connects one channel to the shared gateway process. Alternatively, Snlogin may execute a

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N. W.
WASHINGTON, D.C. 20005
202-408-4000

"privileged process" that determines which channels belongs in the gateway. The privileged process is capable of connecting any channel and is created by a user with access to all channels (e.g., a superuser). This process forwards information from a first socket to a second socket within an address space of the privileged process to establish the gateway. Although methods and systems consistent with the present invention are described as operating in a Unix environment, one skilled in the art will appreciate that such methods and systems can operate in other environments. After the SNlogin script runs in the Supernet context, the SNlogin script spawns a Unix program, such as a Unix shell or a service daemon (step 622). In this step, the SNlogin script spawns a Unix shell from which programs can be run by the user. All of these programs will thus run in the Supernet context until the user runs the SNlogout script.

Fig. 8 depicts a flow chart of the steps performed when sending a packet from node A. Although the steps of the flow chart are described in a particular order, one skilled in the art will appreciate that these steps may be performed in a different order. Additionally, although the SNSL layer is described as performing both authentication and encryption, this processing is policy driven such that either authentication, encryption, both, or neither may be performed. The first step performed is for SNSL layer 542 to receive a packet originating from node A via the TCP/UDP layer and the inner IP layer 540 (step 802). The packet contains a source node ID, a destination node ID, and data. The packet may be received from a process executing in node A connected to a socket. A socket is a well-known software object that connects an application to a network protocol. In UNIX, for example, an application can send and receive TCP/IP messages by opening a socket and reading and

LAW OFFICES
FINNEGAN, HENDERSON, FARABOW, GARRETT, & DUNNER, L.L.P.
1300 I STREET, N.W.
WASHINGTON, D.C. 20005
202-408-4000

writing data to and from the socket. When the packet is received, a Supernet ID and virtual address are appended to a socket structure. The socket structure is modified so as to contain an extra data field for the Supernet ID and virtual address. The addition of a Supernet ID and virtual address in the socket structure enables the Supernet to provide flexible security at a socket level, process level, or node level since the socket structure can discard packets when the sending application/node/process is prohibited from using that channel. Therefore, when a process executing on node A opens a socket to transmit the packet to SNSL layer 542 (step 802), the corresponding Supernet ID and virtual address for that process are also included in the socket request.

The SNSL layer then accesses the VARPDB to obtain the address mapping between the source node ID and the source real address as well as the destination node ID and the destination real address (step 804). If they are not contained in the VARPDB because this is the first time a packet has been sent from this node or sent to this destination, the VARPDB accesses the local VARPD to obtain the mapping. When contacted, the VARPD on the local machine contacts the VARPD that acts as the server for the Supernet to obtain the appropriate address mapping.

After obtaining the address mapping, the SNSL layer determines whether it has been configured to communicate over the appropriate channel for this packet (step 706). This configuration occurs when SNlogin runs, and if the SNSL has not been so configured, processing ends. Otherwise, SNSL obtains the channel key to be used for this channel (step 808). The SNSL maintains a local cache of keys and an indication of the channel to which each key is associated. Each channel key is time stamped to expire in ten seconds, although

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N.W.
WASHINGTON, D.C. 20005
202-408-4000

this time is configurable by the administrator. If there is a key located in the cache for this channel, SNSL obtains the key. Otherwise, SNSL accesses KMD which then locates the appropriate channel key from the appropriate KMC. After obtaining the key, the SNSL layer encrypts the packet using the appropriate encryption algorithm and the key previously obtained (step 810). When encrypting the packet, the source node ID, the destination node ID, and the data may be encrypted, but the source and destination real addresses are not, so that the real addresses can be used by the public network infrastructure to send the packet to its destination.

After encrypting the packet, the SNSL layer authenticates the sender to verify that it is the bona fide sender and that the packet was not modified in transit (step 812). In this step, the SNSL layer uses the MD5 authentication protocol, although one skilled in the art will appreciate that other authentication protocols may be used. Next, the SNSL layer passes the packet to the IP layer where it is then sent to the destination node in accordance with known techniques associated with the IP protocol (step 814).

Fig. 9 depicts a flow chart of the steps performed by the SNSL layer when it receives a packet. Although the steps of the flow chart are described in a particular order, one skilled in the art will appreciate that these steps may be performed in a different order. Additionally, although the SNSL layer is described as performing both authentication and encryption, this processing is policy driven such that either authentication, encryption, both, or neither may be performed. To decapsulate the packet with the additional information (Supernet ID and virtual address), similar to the sending node, the receiving node uses a modified socket structure. The first step performed by the SNSL layer is to receive a packet from the network

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N.W.
WASHINGTON, D.C. 20005
202-408-4000

(step 901). This packet contains a real source address and a real destination address that are not encrypted as well as a source node ID, a destination node ID, and data that are encrypted. Then, it determines whether it has been configured to communicate on this channel to the destination node (step 902). If SNSL has not been so configured, processing ends. Otherwise, the SNSL layer obtains the appropriate key as previously described (step 904). It then decrypts the packet using this key and the appropriate encryption algorithm (step 906). After decrypting the packet, the SNSL layer authenticates the sender and validates the integrity of the packet (step 908), and then it passes the packet to the inner IP layer for delivery to the appropriate node (step 910). To pass the additional information to the other IP layers, the packet is passed using a modified socket structure, as described above. Upon receiving the packet, the inner IP layer uses the destination node ID to deliver the packet.

Fig. 10 depicts a flow chart of the steps performed when logging a node out of a Supernet. The first step performed is for the user to run the SNlogout script and to enter a node ID (step 1002). Next, the SNlogout script requests a log out from SASD (step 1004). Upon receiving this request, SASD removes the mapping for this node from the VARPD that acts as the server for the Supernet (step 1006). SASD then informs KMS to cancel the registration of the node, and KMS terminates this KMC (step 1008). Lastly, KMS generates a new channel key for the channels on which the node was communicating (step1010) to reduce the likelihood of an intruder being able to intercept traffic.

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N.W.
WASHINGTON, D.C. 20005
202-408-4000

Conclusion

Although the present invention has been described with reference to a preferred embodiment, those skilled in the art will know of various changes in form and detail which may be made without departing from the spirit and scope of the present invention as defined in the appended claims and their full scope of equivalents.

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N.W.
WASHINGTON, D.C. 20005
202-408-4000

WHAT IS CLAIMED IS:

1. A method for providing communication access between a first process and a second process comprising the steps, executed in a data processing system, of:

appending security context information for the first process in a process table;

opening a socket between the first process and the second process; and

transmitting a packet from the first process to the second process through the open socket including the security context information for the first process in the process table.

2. The method of claim 1, further comprising modifying a socket structure so as to accept the security context information.

3. The method of claim 1, further comprising:

receiving the packet at the second process through the socket;

verifying the security context information received in the packet; and

permitting use of the packet if the security context information is verified.

4. The method of claim 3, wherein verifying the security context information includes:

determining if the first and second process belong to a channel; and

accepting the transmitted packet when the first and second process belong to the channel.

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N. W.
WASHINGTON, D. C. 20005
202-408-4000

5. The method of claim 4, wherein determining if the first and second process belong to a channel includes:

comparing the security context information in the received packet and security context information in another process table.

6. The method of claim 5, wherein the process table and the another process table are located on a single node.

7. The method of claim 3, wherein verifying the security context information includes:

determining whether the first and second process belong to two different linked channels; and

permitting use of the packet when the different channels are linked.

8. The method of claim 7, wherein determining whether the first and second process belong to two different linked channels includes:

initiating a process that spawns two child processes that are connected by a shared-memory region in a memory.

9. The method of claim 7, wherein permitting use of the packet includes:

decrypting the packet on a node; and

authenticating a sender associated with the first process on the node.

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N. W.
WASHINGTON, D. C. 20005
202-408-4000

10. The method of claim 1, wherein appending security context information includes:

obtaining the security context information from a third process including a virtual address and a node identification; and

limiting each of the first, second and third processes to communicate with another process provided that the communicating processes share the same node identification.

11. The method of claim 1, further comprising:

modifying a network stack such that the network stack requires the security context information to be present in the socket structure to transmit.

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N.W.
WASHINGTON, D.C. 20005
202-408-4000

12. A method for placing processes executed in a node in a security context, comprising the steps of:

sending a request from the node to a server to verify a username and a node identification associated with a process;

in response to the request, receiving security context information at the node from the server including a virtual address for the node;

initiating the process; and

appending the security context information and the node identification associated with the process in a process table.

13. The method of claim 12, wherein receiving security context information further includes:

receiving a key that corresponds to the node identification from the server.

14. The method of claim 13, further comprising:

encrypting a packet transmitted by the process using the key;

encapsulating the encrypted packet with a header that includes the node identification.

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N.W.
WASHINGTON, D.C. 20005
202-408-4000

15. The method of claim 12, further comprising:

sending a second request from the node to the server to verify a username and node identification;

receiving additional security context information from the server, wherein the additional security context information includes a second virtual address for the node;

creating a second process; and

appending the security context information for the second process in the process table that is associated with the second process.

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N. W.
WASHINGTON, D. C. 20005
202-408-4000

16. A method for providing secure communications between a first process and a second process comprising the steps, executed in a data processing system, of:

obtaining a node identification and a virtual address;

including the node identification and the virtual address in a field corresponding to the first process in a process table;

transmitting a datagram that contains the node identification and the virtual address from the first process to a socket; and

receiving the datagram at the second process that contains the node identification and a second virtual address.

17. The method of claim 16, wherein obtaining a node identification and a virtual address further includes:

modifying a socket structure in the socket so that the socket structure accepts the node identification and the virtual address; and

modifying a process table so that the table includes a node identification field and a virtual address field.

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N. W.
WASHINGTON, D.C. 20005
202-408-4000

18. A system for providing communication access between a first process and a second process, comprising:

means for appending security context information for the first process in a process table;

means for opening a socket between the first process and the second process; and

means for transmitting a packet from the first process to the second process through the open socket including the security context information for the first process in the process table.

19. The system of claim 18, further comprising means for modifying a socket structure so as to accept the security context information.

20. The system of claim 18, further comprising:

means for receiving the packet at the second process through the socket;

means for verifying the security context information received in the packet; and

means for permitting use of the packet if the security context information is verified.

21. The system of claim 20, wherein means for verifying the security context information includes:

means for determining if the first and second process belong to a channel; and

means for accepting the transmitted packet when the first and second process belong to the channel.

LAW OFFICES
FINNEGAN, HENDERSON, FARABOW, GARRETT, & DUNNER, L.L.P.
1300 I STREET, N. W.
WASHINGTON, D. C. 20005
202-408-4000

22. The system of claim 21, wherein means for determining if the first and second process belong to a channel includes:

means for comparing the security context information in the received packet and security context information in another process table.

23. The system of claim 22, wherein the process table and the another process table are located on a single node.

24. The system of claim 20, wherein means for verifying the security context information includes:

means for determining whether the first and second process belong to two different linked channels; and

means for permitting use of the packet when the different channels are linked.

25. The system of claim 24, wherein means for determining whether the first and second process belong to two different linked channels includes:

means for initiating a process that spawns two child processes that are connected by a shared-memory region in a memory.

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N. W.
WASHINGTON, D. C. 20005
202-408-4000

26. The system of claim 24, wherein means for permitting use of the packet includes:

means for decrypting the packet on a node; and

means for authenticating a sender associated with the first process on the node.

27. The system of claim 18, wherein means for appending security context information includes:

means for obtaining the security context information from a third process including a virtual address and a node identification; and

means for limiting each of the first, second and third processes to communicate with another process provided that the communicating processes share the same node identification.

28. The system of claim 18, further comprising:

means for modifying a network stack such that the network stack requires the security context information to be present in the socket structure to transmit.

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N. W.
WASHINGTON, D.C. 20005
202-408-4000

29. A system for placing a process executed in a node in a security context, comprising:

a server; and

a sending node comprising:

a transmission module that transmits a request to the server to verify a user name and a node identification, and receives security context information from the server in response to the request, wherein the security context information includes a virtual address for the sender node;

memory containing a process and an associated process table; and

an appending module that appends the received security context information and the node identification for the process in the process table.

30. The system of claim 29, wherein the transmission module further receives a key that corresponds to the node identification from the server.

31. The system of claim 30, further comprising:

an encryption module that encrypts a packet transmitted by the process using the key;

an encapsulating module that encapsulates the encrypted packet with a header that includes the node identification.

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N.W.
WASHINGTON, D.C. 20005
202-408-4000

32. The system of claim 29, further comprising:

a gateway that provides communication between the process and a second process executing in the node, and

wherein the transmission module further sends a second request to the server to verify a username and node identification, and receives additional security context information from the server, wherein the additional security context information includes a second virtual address for the node;

appending the security context information for the second process in a process table that is associated with the second process.

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N. W.
WASHINGTON, D. C. 20005
202-408-4000

33. A system for providing secure communications between a first process and a second process, comprising:

means for obtaining a node identification and a virtual address;

means for including the node identification and the virtual address in a field corresponding to the first process in a process table;

means for transmitting a datagram that contains the node identification and virtual address from the first process to a socket; and

means for receiving the datagram at the second process that contains the node identification and a second virtual address.

34. The system of claim 33, wherein means for obtaining a node identification and a virtual address further comprises:

means for modifying a socket structure in the socket so that the socket structure accepts the node identification and the virtual address; and

means for modifying a process table so that the table includes a node identification field and a virtual address field.

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N.W.
WASHINGTON, D.C. 20005
202-408-4000

35. A computer readable medium for controlling a data processing system to perform a method for providing communication access between a first process and a second process, comprising:

an appending module for appending security context information for the first process in a process table;

an opening module for opening a socket between the first process and the second process; and

a transmitting module for transmitting a packet from the first process to the second process through the open socket including the security context information for the first process in the process table.

36. The computer readable medium of claim 35, further comprising a modifying module for modifying a socket structure so as to accept the security context information.

37. The computer readable medium of claim 35, further comprising:

a receiving module for receiving the packet at the second process through the socket;

a verifying module for verifying the security context information received in the packet; and

a permitting module for permitting use of the packet if the security context information is verified.

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N. W.
WASHINGTON, D. C. 20005
202-408-4000

38. The computer readable medium of claim 36, wherein the verifying module includes:

a determining module for determining if the first and second process belong to a channel; and

an accepting module for accepting the transmitted packet when the first and second process belong to the channel.

39. The computer readable medium of claim 38, wherein the determining module includes:

a comparing module that compares the security context information in the received packet and security context information in another process table.

40. The computer readable medium of claim 39, wherein the process table and the another process table are located on a single node.

41. The computer readable medium of claim 37, wherein the verifying module includes:

a determining module for determining whether the first and second process belong to two different linked channels; and

a permitting module for permitting use of the packet when the different channels are linked.

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N.W.
WASHINGTON, D.C. 20005
202-408-4000

42. The computer readable medium of claim 41, wherein the determining module includes a initiating module that initiates a process that spawns two child processes that are connected by a shared-memory region in a memory.

43. The computer readable medium of claim 41, wherein the permitting module includes:

a decrypting module for decrypting the packet on a node; and

an authenticating module for authenticating a sender associated with the first process on the node.

44. The computer readable medium of claim 35, wherein the appending module includes:

an obtaining module for obtaining the security context information from a third process including a virtual address and a node identification; and

a limiting module for limiting each of the first, second and third processes to communicate with another process provided that the communicating processes share the same node identification.

45. The computer readable medium of claim 35, further comprising:

a modifying module for modifying a network stack such that the network stack requires the security context information to be present in the socket structure to transmit

LAW OFFICES
FINNEGAN, HENDERSON,
FARABOW, GARRETT,
& DUNNER, L.L.P.
1300 I STREET, N. W.
WASHINGTON, D. C. 20005
202-408-4000

ABSTRACT

Methods and systems consistent with the present invention provide dynamic security policies that change the granularity of the security at the node level, process level, or socket level. Specifically, a channel number and virtual address are associated with various processes included in a process table. Since a security policy is required for all processes, secure and insecure processes located on the same channel may communicate with one another. Moreover, processes located on different channels may communicate with one another by a gateway that connects both channels. This scalable blanketing security approach provides an institutionalized method for securing any process, node or socket by providing a unique mechanism for policy enforcement at runtime or by changing the security policies.

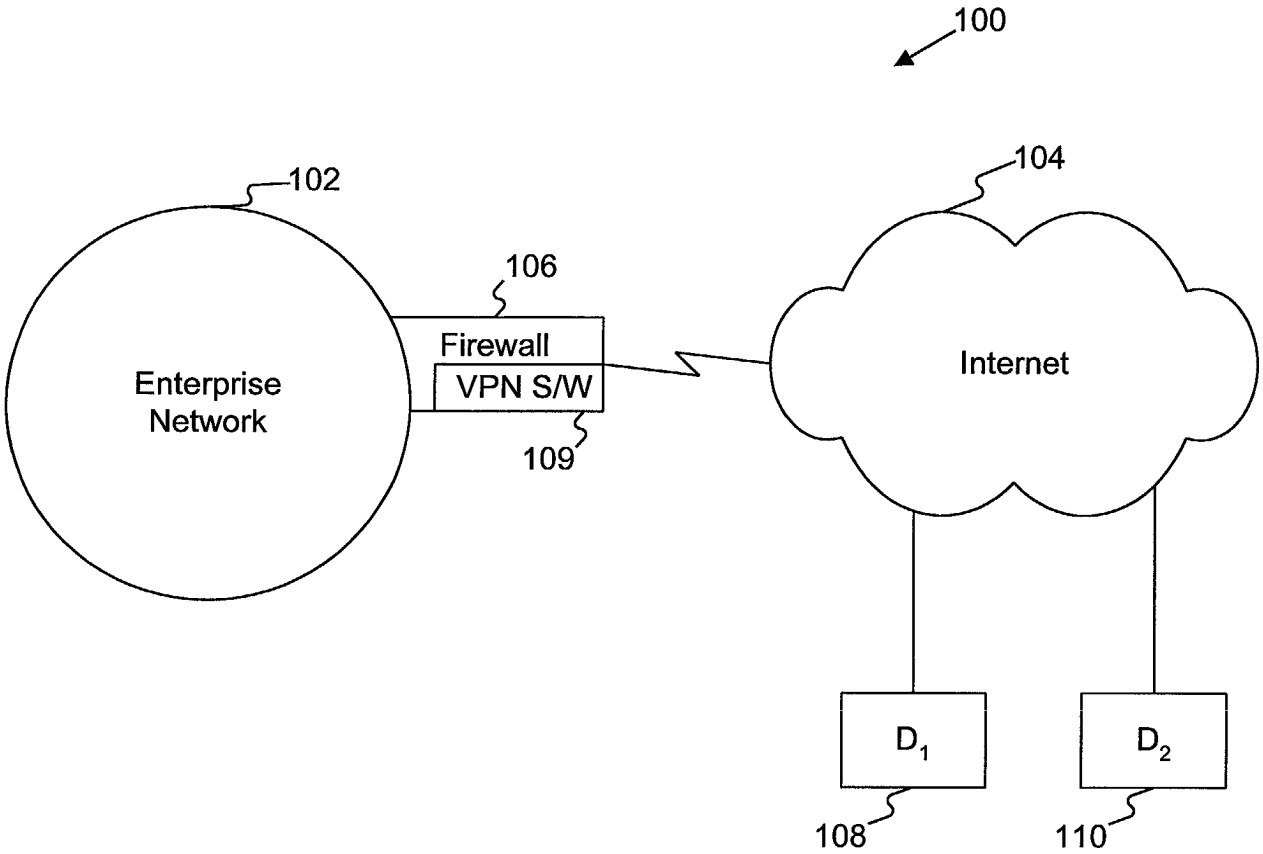

Fig. 1
(Prior Art)

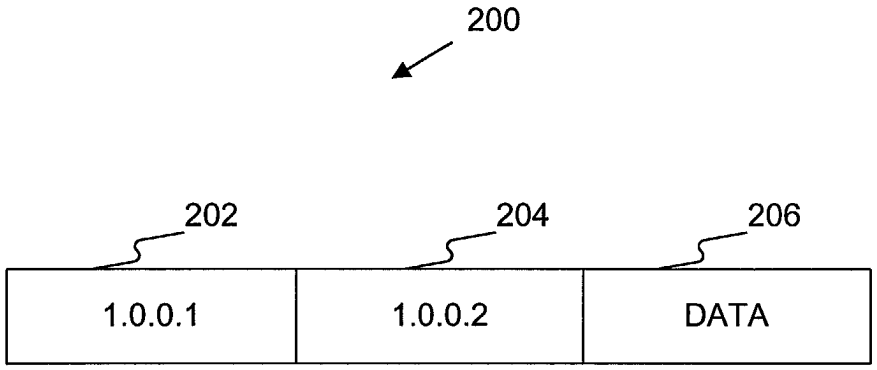

Fig. 2A
(Prior Art)

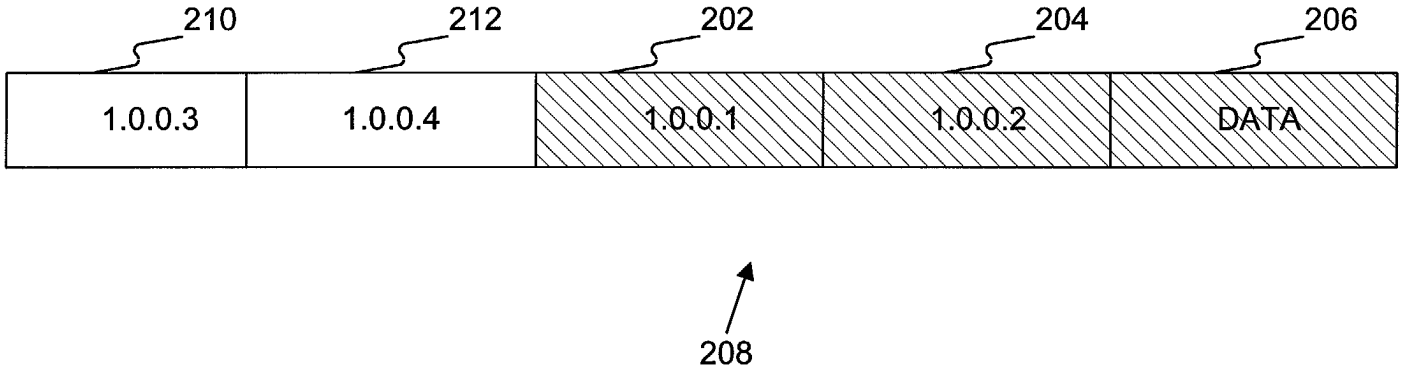

Fig. 2B
(Prior Art)

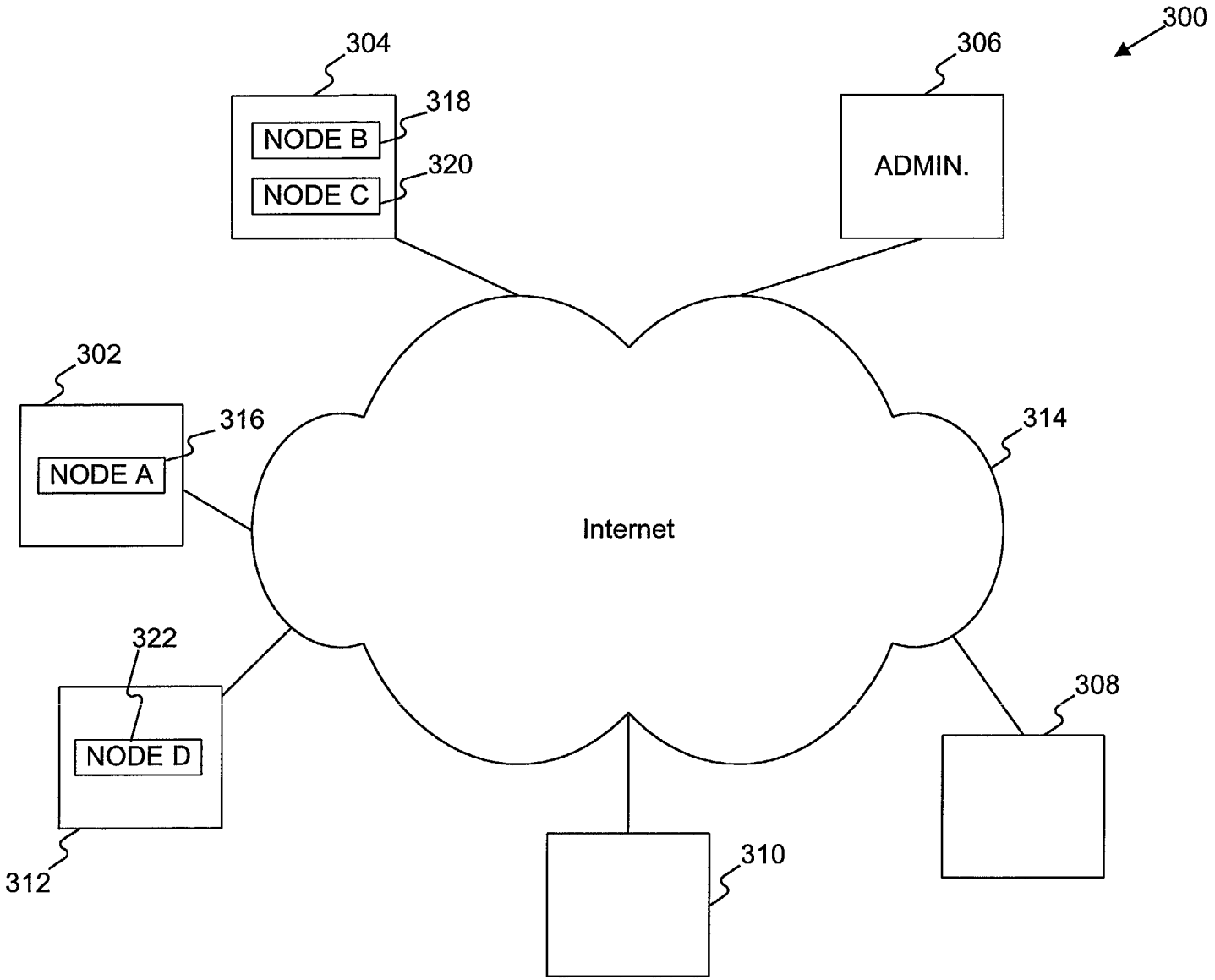
300
304
306
318
NODE B
320
NODE C
ADMIN.
302
316
NODE A
314
Internet
322
NODE D
308
312
310

Fig. 3

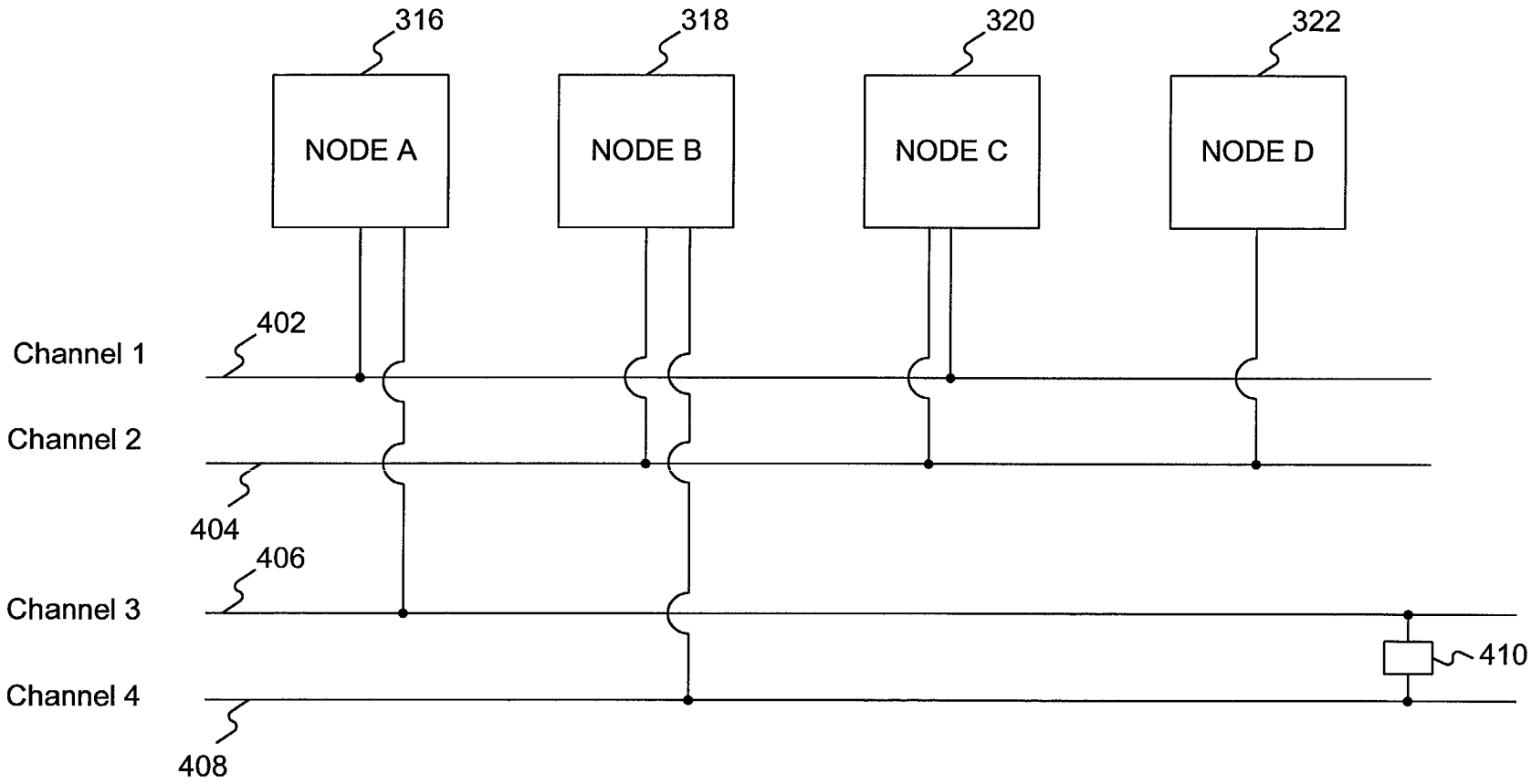
316
318
320
322
NODE A
NODE B
NODE C
NODE D
402
Channel 1
Channel 2
404
406
Channel 3
410
Channel 4
408

Fig. 4

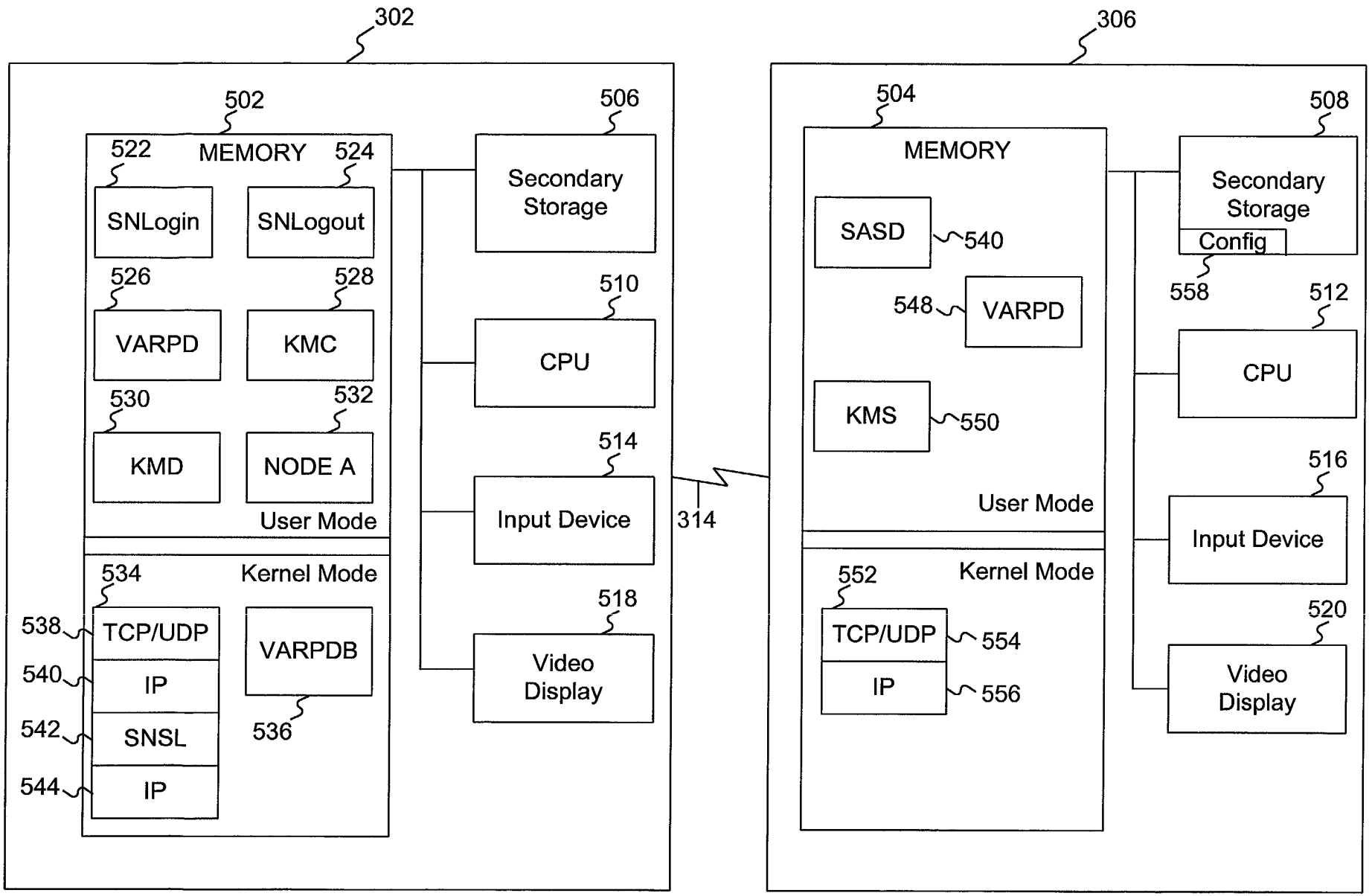
302
306
502
MEMORY
522
524
SNLogin
SNLogout
526
528
VARPD
KMC
530
532
KMD
NODE A
User Mode
534
Kernel Mode
538
TCP/UDP
540
IP
542
SNSL
544
IP
VARPDB
536
506
Secondary Storage
510
CPU
514
Input Device
518
Video Display
314
504
MEMORY
SASD
540
548
VARPD
KMS
550
User Mode
552
Kernel Mode
TCP/UDP
554
IP
556
508
Secondary Storage
Config
558
512
CPU
516
Input Device
520
Video Display

Fig. 5

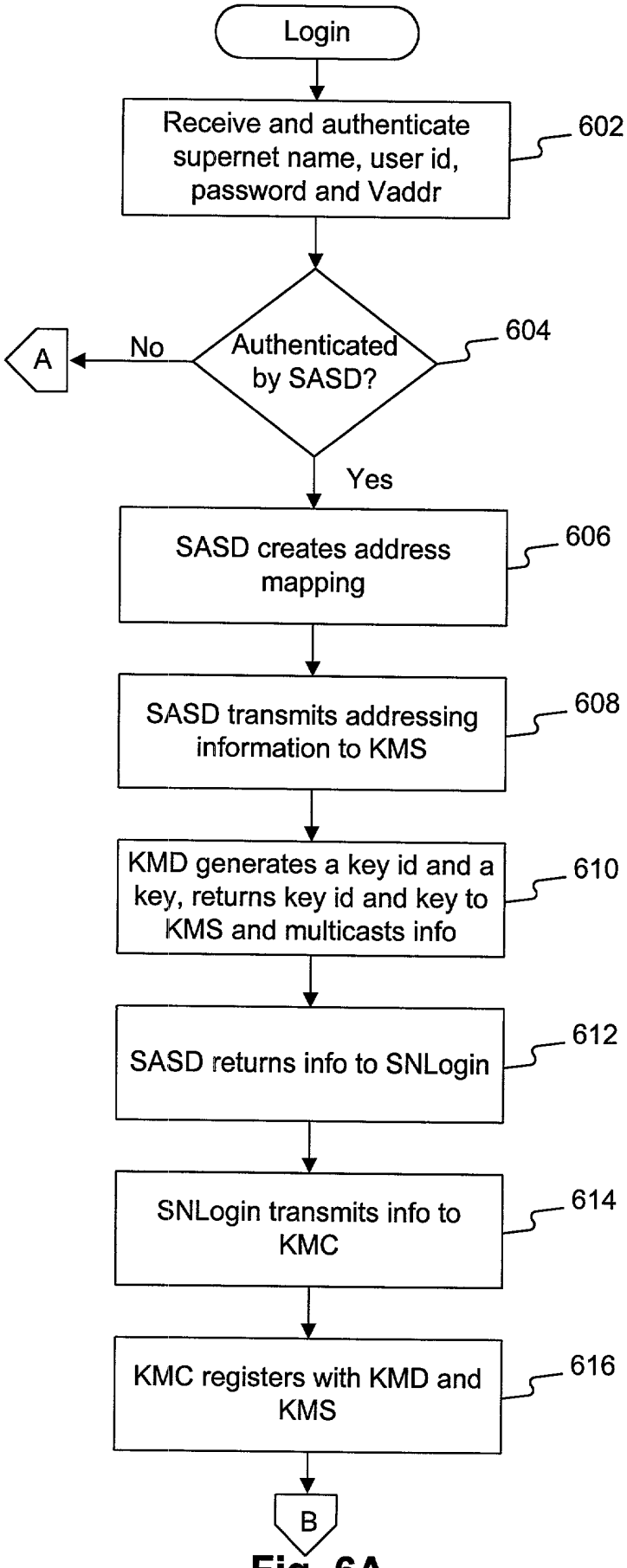
Login
Receive and authenticate supernet name, user id, password and Vaddr
602
Authenticated by SASD?
604
No
A
Yes
SASD creates address mapping
606
SASD transmits addressing information to KMS
608
KMD generates a key id and a key, returns key id and key to KMS and multicasts info
610
SASD returns info to SNLogin
612
SNLogin transmits info to KMC
614
KMC registers with KMD and KMS
616
B

Fig. 6A

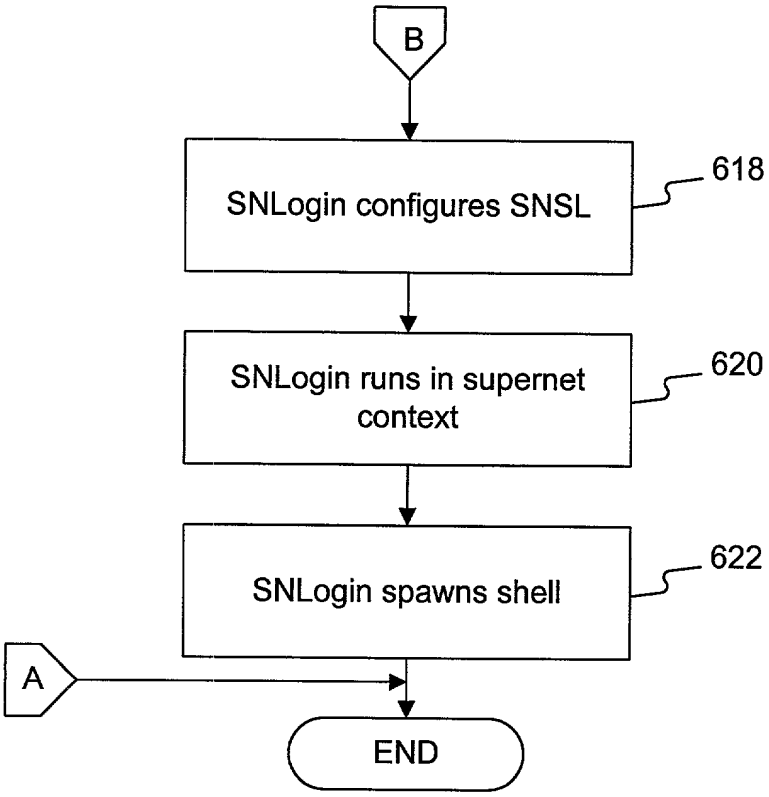
B
SNLogin configures SNSL
618
SNLogin runs in supernet context
620
SNLogin spawns shell
622
A
END

Fig. 6B

700

750

PROCESS TABLE				
710	720		730	740
Process ID	Process Descriptor	. . .	Supernet ID	VADDR
9	FTD	. . .	0 x 123	10.0.0.1
8	HTTD	. . .	0 x 123	10.0.0.2
			0 x 123	10.0.0.1

Fig. 7

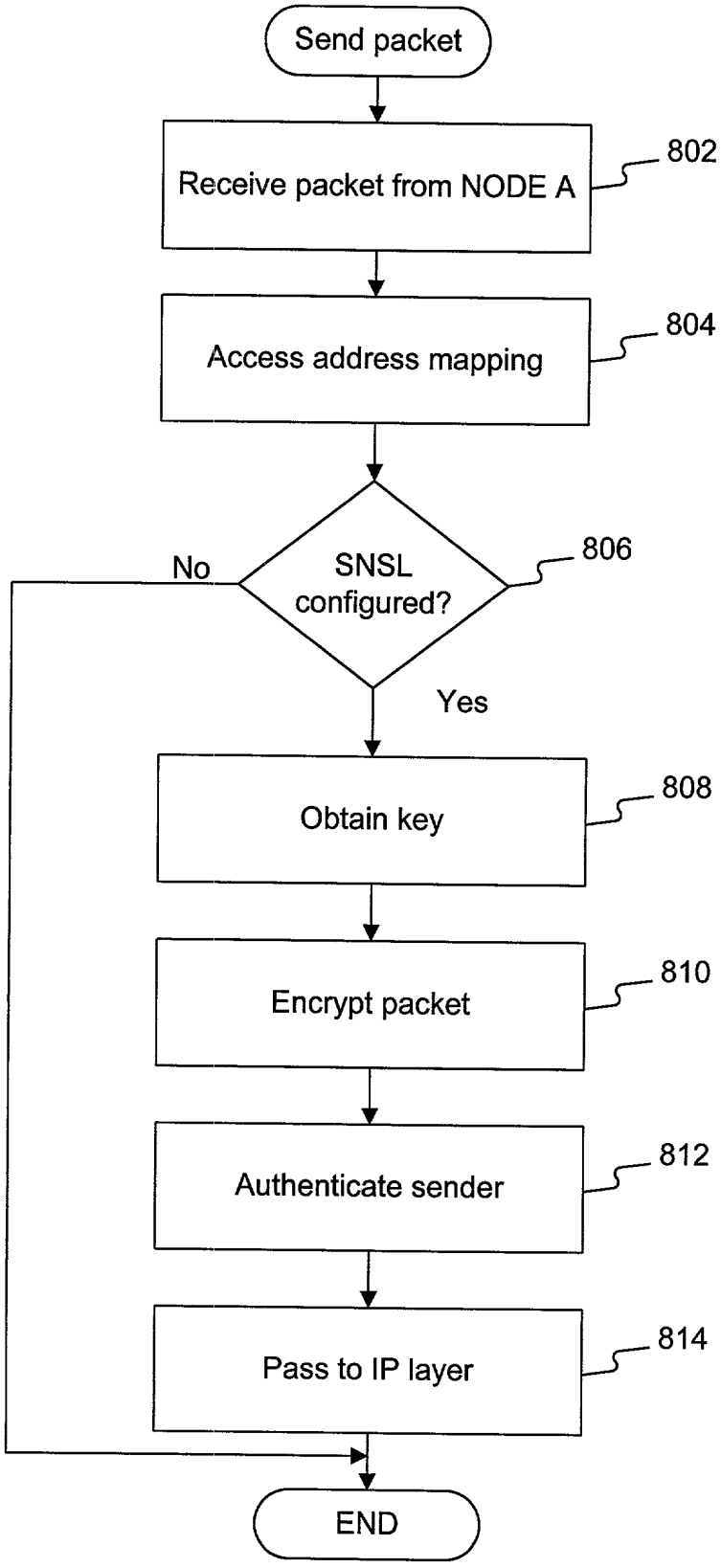
Send packet
Receive packet from NODE A
802
Access address mapping
804
SNSL configured?
806
No
Yes
Obtain key
808
Encrypt packet
810
Authenticate sender
812
Pass to IP layer
814
END

Fig. 8

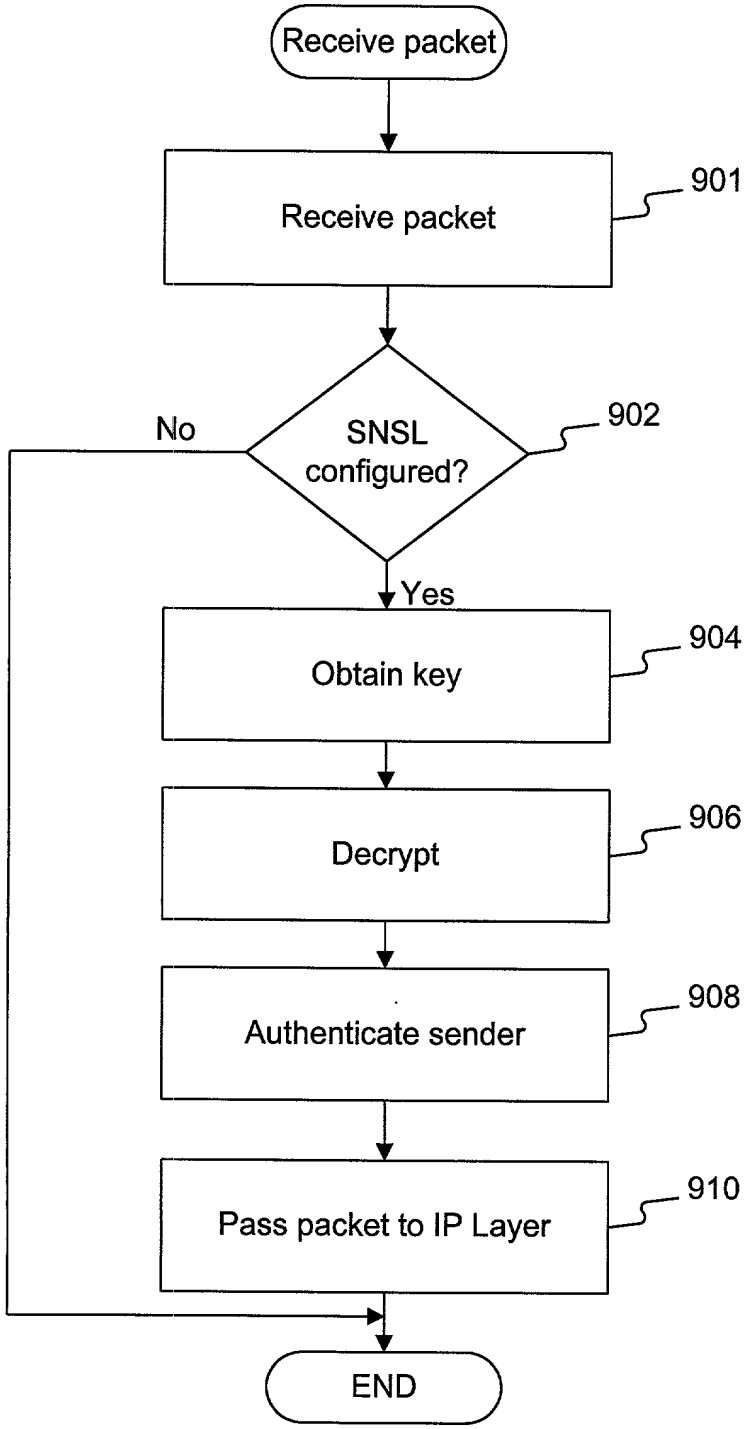
Receive packet
Receive packet
901
SNSL configured?
902
No
Yes
Obtain key
904
Decrypt
906
Authenticate sender
908
Pass packet to IP Layer
910
END

Fig. 9

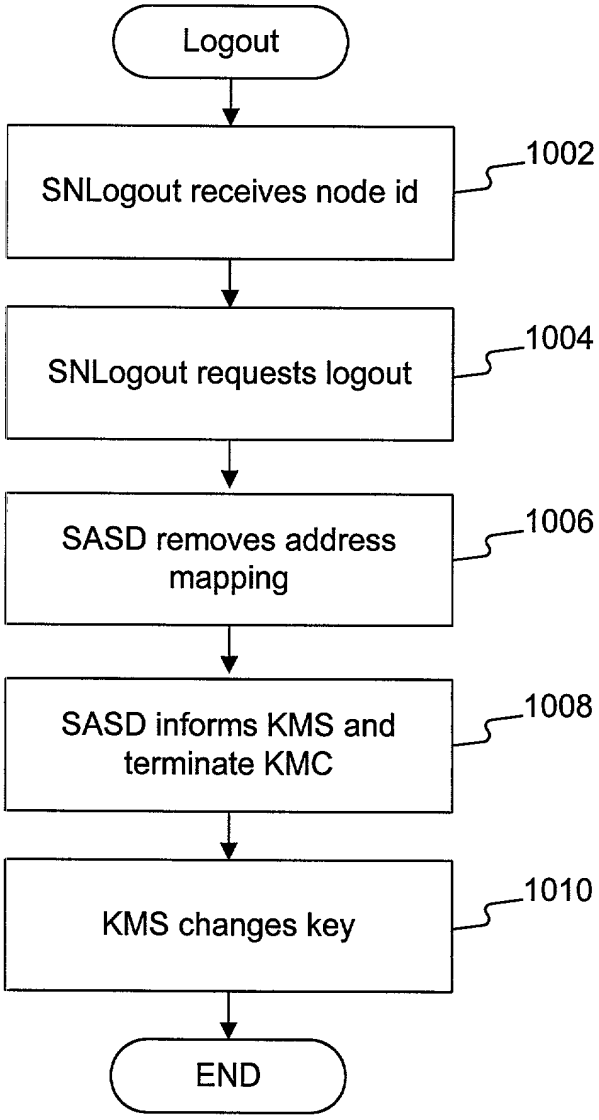
Logout
SNLogout receives node id
1002
SNLogout requests logout
1004
SASD removes address mapping
1006
SASD informs KMS and terminate KMC
1008
KMS changes key
1010
END

Fig. 10